

Multi-perspective Automated Analysis

Robert A. Martin
Sean Barnum

May 2011

Report Documentation Page				Form Approved OMB No. 0704-0188	
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.					
1. REPORT DATE MAY 2011		2. REPORT TYPE		3. DATES COVERED 00-00-2011 to 00-00-2011	
4. TITLE AND SUBTITLE Multi-perspective Automated Analysis				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) The MITRE Corporation, 202 Burlington Rd, Bedford, MA, 01730-1420				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited					
13. SUPPLEMENTARY NOTES Presented at the 23rd Systems and Software Technology Conference (SSTC), 16-19 May 2011, Salt Lake City, UT. Sponsored in part by the USAF. U.S. Government or Federal Rights License					
14. ABSTRACT					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT Same as Report (SAR)	18. NUMBER OF PAGES 27	19a. NAME OF RESPONSIBLE PERSON
a. REPORT unclassified	b. ABSTRACT unclassified	c. THIS PAGE unclassified			

Agenda

**8:00-8:45am Software Security Knowledge
about Applications Weaknesses**

**9:00-9:45am Software Security Knowledge
about Attack Patterns Against
Applications**

Training in Software Security

10:15-11:00am Software Security Practice

11:15-12:00am Supporting Capabilities

Assurance Cases

**Secure Development & Secure
Operations**

Key Role of Application Security Risk Analysis in the Cybersecurity Game

- **Ultimate goal is to prevent security vulnerabilities from ever entering software**
- **Reality is they are already there and even new code from security-aware developers needs to be checked**
- **Application security risk analysis is the practice of:**
 - checking software for weaknesses/vulnerabilities
 - characterizing the risk they pose
 - identifying and prioritizing mitigations

Varying Perspectives of Analysis

- static source code
- static binary code
- dynamic application scanning
- application penetration testing
- application data security
- fuzzing
- complexity
- composition & pedigree
- etc.

Varying Capabilities of Analysis Perspectives

- Different perspectives are effective at finding different types of weaknesses
- Some are good at finding the cause and some at finding the effect

	Static Code Analysis	Penetration Test	Data Security Analysis	Code Review	Architecture Risk Analysis
Cross-Site Scripting (XSS)	X	X		X	
SQL Injection	X	X		X	
Insufficient Authorization Controls		X	X	X	X
Broken Authentication and Session Management		X	X	X	X
Information Leakage		X	X		X
Improper Error Handling	X				
Insecure Use of Cryptography		X		X	X
Cross Site Request Forgery (CSRF)		X		X	
Denial of Service	X	X	X		X
Poor Coding Practices	X			X	

Automating Analysis Perspectives

- **Automation should be leveraged wherever possible but should be combined with focused manual analysis**
- **Automated tools will find the low-hanging fruit much faster than manual analysis can**
- **Manual analysis will find less obvious and occasionally high-risk issues**

Current State of the Practice

- **Most organizations undertaking application security risk analysis only perform one or maybe two analysis perspectives and those are done as independent processes often by separate teams**
 - If developer-centric organization, typically start with static analysis
 - If test-centric, typically start with application scanning and penetration testing
 - If information assurance or data-centric, typically start with data security scanning

The Gestalt of Multi-perspective Analysis

- Better situational awareness
- Reinforce confidence in findings of each perspective
- Combine the assurance of dynamic analysis with the detail of structure analysis to plan effective mitigation of high-criticality risk

The Challenges of Integrated Multi-perspective Analysis

- Varying perspectives have different drivers and priorities based on context
- Differing perspectives treat “location” of issue differently making correlation a challenge
- Each tool for each perspective has its own reporting schema
 - Need for a unified findings schema (SAFES)

The Need for Standards in Effective Integration

- **Always make sure comparing apples to apples**
- **Weakness**
 - Common Weakness Enumeration (CWE)
- **Attack**
 - Common Attack Pattern Enumeration and Classification (CAPEC)
- **Vulnerability**
 - Common Vulnerabilities and Exposures (CVE)
- **Technical Context**
 - Common Platform Enumeration (CPE)
- **Mitigation**
 - Common Control Enumeration (CCE)

A Recommended Baseline for Multi-perspective Analysis

- **To effectively assess the security risk of an application, an assessment methodology should at a minimum include the following perspectives:**
 - Static source code analysis
 - Application scanning & penetration testing
 - Application data security analysis

Static Source Code Analysis

- **Analyze code without executing it**

- **Strengths**

- Fast compared to manual code review
- Fast compared to testing
- Complete, consistent coverage of source code (all paths)
- Brings security knowledge with it

- **Limitations**

- Only analyzes the source code you feed it
- Doesn't find everything
 - Architecture errors
 - Bugs you're not looking for
 - System administration mistakes
 - User mistakes
- False positives

- **Multi-perspective integration value**

- Actual location of the weakness in code
- Identify issues to target with penetration testing
- Identify co-influencing weaknesses within relevant contexts

Application Scanning & Penetration Testing

- **Security testing (black box) of applications through simulated attacks**
- **Strengths**
 - Simulates the actual risk (attacker's action)
 - Tests full software stack
 - Low false positives
 - Mature technology
- **Limitations**
 - Only as good as what you scan (crawling limitations)
 - Analysis limited to the test cases executed
 - Must run tests often to stay protected
 - Can only be performed once code is 'runable'
 - Risky to run on production applications
 - Cannot identify the actual source of the problem, only the symptom
- **Multi-perspective integration value**
 - Confirming that weaknesses are vulnerable
 - Mapping penetration scans to locations in source code
 - Mapping data security findings to injection findings, privilege issues, etc.

Application Data Security Analysis

- **Analyzing the security concerns of how an application accesses and manages its database**
- **Strengths**
 - Analyzes a live, fully configured system rather than just source code
 - Good at catching really bonehead mistakes (they are more common than you think)
 - Helps mitigate both insider and external threats
- **Limitations**
 - Only as good as what you tell it to look for
 - Does not understand semantics of data (can use limited proxies)
- **Multi-perspective integration value**
 - Confirmation of likely weaknesses as vulnerabilities
 - Better contextual info about nature and severity of weaknesses
 - Improved understanding of likelihood of weaknesses being exploitable
 - Increases accuracy of forensic data
 - Improved data flow policies
 - Improved Access Control

Summary and Conclusions

- **Software Assurance analysis is increasingly becoming a high priority and is maturing in its capability**
- **Varying perspectives of analysis are available, each with their own unique value**
- **Blending multiple perspectives together yields better overall coverage and an integrated gestalt**
- **It is real and possible to begin pursuing this approach today**

Taming the Tower of Babel: Software Assurance Findings Expression Schema (SAFES)



Sean Barnum
MITRE



**Homeland
Security**

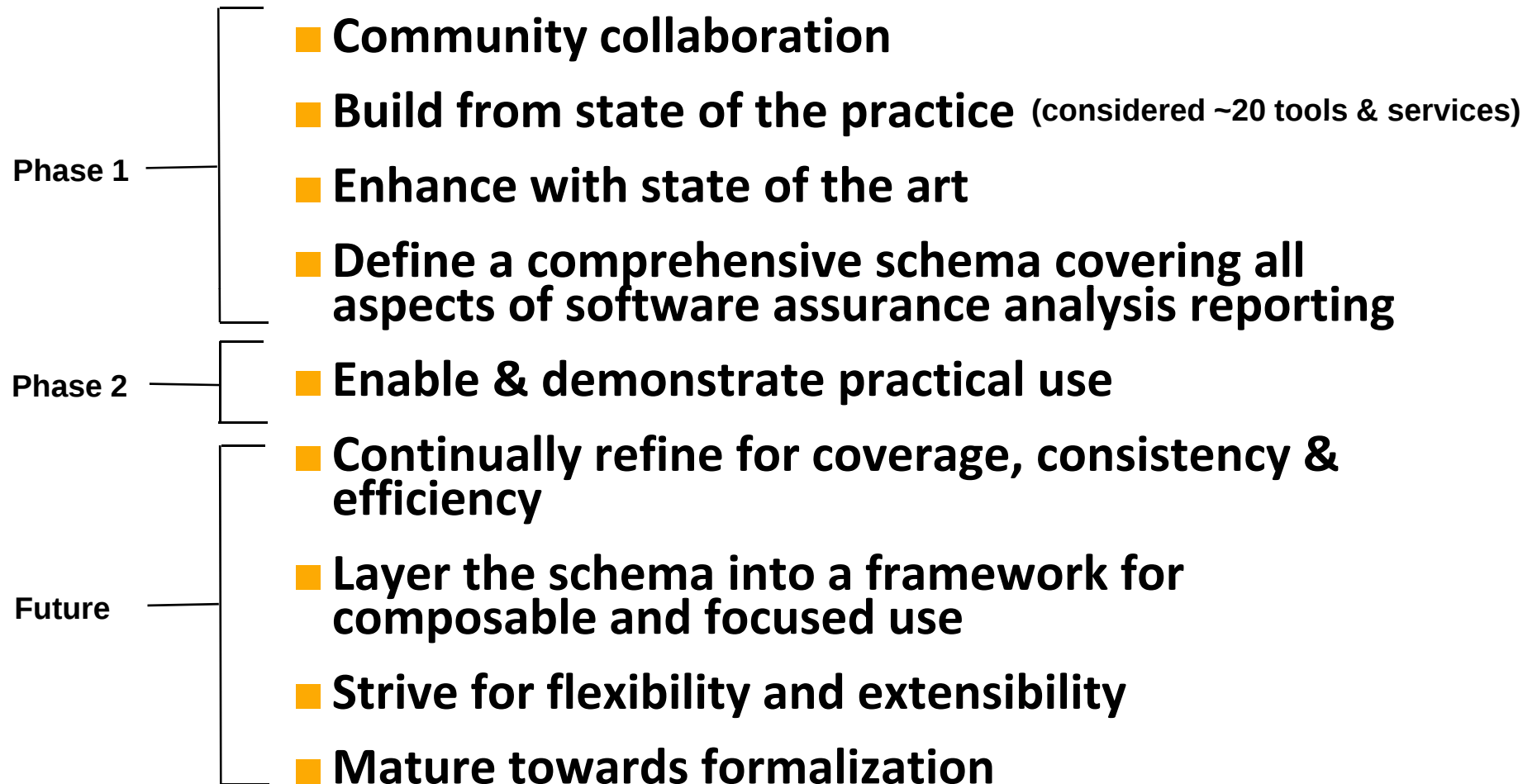
Today's Challenge

- **There is no standard reporting format for SwA analysis**
 - Very difficult to combine results of multi-perspective analysis
 - Very difficult to combine results of multi-tool analysis
 - Very inefficient for tool vendors looking to integrate results with other tools (very costly and redundant)
 - Very difficult to trend across assessments from different tools or analysts
 - Very difficult to automate meta-analysis and the assessment process

SAFES Effort

- **Software Assurance Findings Expression Schema (SAFES)**
- **Sponsored by the NSA Center for Assured Software (CAS)**
- **Objectives:**
 - Enable and encourage consistency in software assurance tool, service and analysis practice findings
 - Establish more structured and effectively useful software assurance tool, service and analysis practice results
 - Enable integration of results from multiple software assurance tools, services or analysis practices
 - Enable automated processing of software assurance tool, service or analysis practice results

SAFES Approach



SAFES Initial Scope

■ In-scope perspectives for initial effort:

- Static source code analysis
- Static binary code analysis
- Web application penetration testing
- Data security analysis
- Fuzzing
- Threat modeling
- Architectural risk analysis

■ Some vendors actively collaborating others were passively incorporated

SAFES is a comprehensive and detailed schema

■ **Info on findings**

- Description
- Categorization
- Location
- Prioritization
- Correlations

■ **Info on analysis approach**

- Tool or service
- Methodology
- Detection mechanisms

- **Info on mitigation**
- **Info on meta-analysis**
- **Info on personnel**
- **Info on application**
 - Structure, content & configuration
 - Business/mission and security context
- **Info on assurance case**
- **Info on threat analysis**

Key Constructs

- **Sub-Assessment scopes**
- **Traces**
- **Report views**
- **Assurance case**
- **Finding prioritization**
- **Tool-Service info**
- **Findings correlations**

A Sampling of Potential Use Cases

- Understand the Business Context of application
- Identify risks
- Map technical risks to business context
- Map the application attack surface
- Identify relevant threats
- Inventory and characterize assets
- Create threat model
- Define FISMA security categorization (FIPS-199)
- FISMA Security Planning (SP800-18)
- FISMA Risk Assessment (SP800-30)
- Conduct multi-tool/multi-perspective analysis
- Identify false positives
- Characterize risk
- Prioritize risk
- Correlate findings
- Stitch dynamic & static location results
- Integrate automated and manual analysis
- Reuse common mitigation advice
- Create assessment report
- Create different versions of report
- Define an assurance case for an application
- Create an assurance case compliance report
- Import CWE content into local context
- Identify common finding trends across portfolio by technology context
- Maintain analysis accountability
- Identify trends in tool and rule efficacy
- Mapping between various tool level definitions

SAFES Maturation Paths

- **Usability:** primarily focused on efforts surrounding the schema to make it more usable by the community such as native transforms, tooling, etc.
- **Refinement:** primarily focused on improving the quality and coverage of the schema itself with activities such as adding new perspectives, adding new schemas, fixing errors, etc.
- **Formalization:** primarily focused on gradually (as quickly as is prudent and accepted by the targeted user community) incorporating in formal standards-based approaches (vocabulary, structure, etc.) and working towards handoff of development to an appropriate community standards consortium body

SAFES Phase 2

- **Develop 5-10 transforms from native tool output to SAFES (currently for CAS internal use but hopefully will eventually be shared)**
- **Develop a demonstrative use case example for SAFES**
- **Develop lightweight initial prototype authoring/editing/reporting tools (very, very simple)**
- **Develop a real, permanent website as part of MSM**
- **Coordinate with standards organizations for planning towards future maturation and formalization**

SAFES Next Steps Beyond Phase 2

- **Identify & support real-world prototype usage of SAFES**
- **Refine based on feedback**
- **Refine & extend authoring/editing/reporting tools with the goal of eventually transferring this work to other parties (vendors, open-source projects, consortia, etc.)**
- **Incorporate coverage for more tools, services & analysis practices**
- **Work with vendors (and OS projects) to develop more native transforms and encourage native output of SAFES**
- **Refine for efficiency**
- **Refine for flexibility (framework layering)**
- **Refine for formalization towards existing standards**

Questions?

Sean Barnum
MITRE
sbarnum@mitre.org

